Go Live Checklist

Preparing to Go Live

January 2009
Version G.A.

FatWire Software
330 Old Country Road
Mineola, NY 11501
516.328.9473
www.fatwire.com

FatWire®
SOFTWARE

Go Live Checklist

1 Ensure that publishing is working correctly 3
- 1.1 Analyze Firewall 3
- 1.2 Ensure that all information published is correct 3
- 1.3 Review Page Cache 3
- 1.4 Review batch host configuration 4

2 Load Testing 4
- 2.1 Load Test Delivery 4
- 2.2 Verify cluster configuration 4
- 2.3 Simulate Editorial Load 5

3 Performance Testing 5
- 3.1 Performance Test Publishing 5
- 3.2 Performance Test Content Server 5
- 3.3 Performance Test additional components eg file system, Engage 6

4 Perform a security audit of all external-facing environments 6
- 4.1 Review Passwords 6
- 4.2 Harden servlets 7
- 4.3 Harden non-delivery pages 7
- 4.4 Protect against remote calls 8

5 Additional standard tests 8
- 5.1 Review filesystem configuration 8
- 5.2 Review Database configuration and response times 9
- 5.3 Review and assess total number of file-descriptors 9
- 5.4 Perform sizing on the java heap size 10
- 5.5 Review Page caching strategy 10

6 Backup Testing 10
- 6.1 Test database backups 10
- 6.2 Test application backups 10
- 6.3 Test application server backups 11
- 6.4 Test disaster recovery 11

7 Information transfer 11
- 7.1 Provide go-live implementation plan to support 11
- 7.2 Document all customizations to product behavior, eg UI elements, modified jars 11
- 7.3 Document all known outstanding issues and any action plan for their resolution 11
- 7.4 Supply names/contact details of persons responsible for managing the customer go-live 11
- 7.5 Provide all design documentation describing site design, asset models etc 11
- 7.6 Provide architectural map and details such as CS versions, database, webservers, load-balancers, etc. 11
- 7.7 Provide an export of database, FatWire directory, Shared Filesystem and web application to support 11
- 7.8 Confirm completion (or otherwise) and results of all items above 11

Appendix A: Checklist 12

1 Ensure that publishing is working correctly

At least one publishing test must be completed from your management environment to your delivery environment.

1.1 Analyze Firewall

Check Firewall ports and timeouts for the connection between source and destination. Content Server requires communications on different ports between different instances.

For the development server(s) they will need to have the firewall opened for the ports specified in your publishing destinations. If you're running a cluster, the ports for the batchhost and eventhost need to be setup on the firewall. If you're using a HA setup that requires for traffic to pass through a Firewall for localhost traffic, you will need to open these locally as well.

For the management servers, you will need to configure exactly as listed above for the development servers but for management.

For the delivery servers, you will need to make sure that your webservers communication ports have been allowed on the firewall.

For each of your different environments you should check the actual firewall and router setup rules to make sure that sockets are not forcibly disconnected after a period of time.

1.2 Ensure that all information published is correct

Perform a complete publish and compare the environments to make sure that all content was published correctly.

1.3 Review Page Cache

Ensure page cache is operating as intended; content is flushing when assets are published.

Check your system page cache and system item cache before a publish and record its contents. After a publish check the assets that were published to ensure that they have been updated in the system page cache and system item cache.

You can also turn on debugging for publishing on the host to ensure that the correct IDs were flushed.

1.4 Review batch host configuration

Ensure batch host is set correctly on source and destination.

The batch host should be set to one server on each of the instances, for example if you're running publishes on your management instance you should set the batch host on each member of the cluster to be one member.

The properties for batchhost can be located in the futuretense_xcel.ini, the properties are;
xcelerate.batchhost
xcelerate.batchuser
xcelerate.batchpass

2 Load Testing

A load test should be performed on all parts of your environment to ensure that the current architecture setup meets the metrics defined for FatWire.

2.1 Load Test Delivery

Comprehensive load test of delivery system, all pages with simulated load no less than the predicted amount of page views.

Calculate based on current trends the amount of pageviews that your site current is undergoing. From those figures setup a Jrun, LoadRunner etc script to simulate the usage patterns of your users against your site. Ensure that the load test is a comprehensive test that covers the actual pattern of usage and not just one page. During the load test record statistics on each of your different components, the components that you should monitor are; Database (CPU, Memory, Shared Memory, Cursors), Application Server(CPU, Memory, JVM Heap Space, File Handles), Webservers (CPU, Memory, Sockets, File Handlers).

2.2 Verify cluster configuration

Ensure that ft.sync and ft.usedisksync is enabled even if no cluster is present, so that any issues with shared disk are discovered early and do not present a surprise when embarking on a cluster later.

Check your futuretense.ini files to make sure that you have settings for the ft.sync and the ft.usedisksync properties. If you need information about each of these properties refer to the property guide available on e-docs.

2.3 Simulate Editorial Load

Simulate load on staging system, eg expected number of editorial users all performing normal tasks.

Perform load testing on your staging system by using either people or an automated testing tool (JRun, LoadRunner, etc) to simulate your editorial load on the system. During this load testing you should monitor the statistics of your components, eg Database (CPU, Memory, Shared Memory, Cursors), Application Server(CPU, Memory, JVM Heap Space, File Handles), Webservers (CPU, Memory, Sockets, File Handlers)

3 Performance Testing

You should complete performance testing on your environment so that it meets your metrics.

3.1 Performance Test Publishing

Performance testing should be similar to load testing but with a different outcome in mind. When you performance test your publishing procedures you should monitor the same components as listed in the load testing however you should address the areas of each of these components that are performing slowly.

When performance testing publishing you should turn on several different debug logging options, they are; publishing debug, time debug, cs debug and db debug. These four debugs will allow you to see the different activities undertaken during the publishing process and will allow you to tune the different areas of your Content Server installation with publishing in mind.

3.2 Performance Test Content Server

To performance test Content Server you should turn on three different debugs, cs debug, db debug and time debug. These will allow you to see how Content Server is performing under normal operating scenarios. Ensure that no publishing activity is occurring during normal Content Server testing. Based on the log files you can address different areas of your setup that are performing slowly.

3.3 Performance Test additional components eg file system, Engage

Any additional component that your server requires outside of the normal Content Server installation should be performance tested. Some common areas that need to be analyzed and corrected are;
File Systems on shared cluster instances.
Engage tables if engage is being used.
Database performance.
Network traffic performance.

4 Perform a security audit of all external-facing environments

4.1 Review Passwords

Ensure all passwords have been changed from the default users.

Many passwords are defaulted to different common passwords, for example fwadmin is usually defaulted to xceladmin. Review all default passwords to ensure that people with prior Content Server experience can not login or perform unwanted actions on your installation.

You should review all entries in your systemusers table to ensure that any user that has the sitegod privilege changes their password on a regular basis.

The list of usernames that you should change is;
fwadmin
SatelliteServer

4.2 Harden servlets

Only the necessary servlets are exposed to the world, if all access goes through Satellite Server then only Satellite and CookieServer servlets need to be exposed.

Ensure that your webservers are setup correctly to only allow access to the parts of the servlet that you need to provide pages. For example, if you're going to be serving all your content from Satellite Server, then make sure that you block the Content Server servlet.

The complete list of servlets to block against is;
HelloCS
CatalogManager
TreeManager
DebugServer
CacheServer
Inventory
PageDispatcherServer
DispatchManager
SyncSeedDispatchServer
SeedDispatchServer
FlushServer

4.3 Harden non-delivery pages

Disable outside access to all non-delivery pages such as support tools, UI pages, WebServices.

If any of the non delivery pages are installed on your delivery system make sure that you have the correct settings in your webserver to deny outside access to these pages. You should set rules on your webserver to deny access to these specific pages externally and only allow internal traffic to these webpages. From support tools you can perform a variety of operations that can damage your system if used improperly therefore it should be restricted to internal users only.

4.4 Protect against remote calls

Either with webserver rules, or by setting csstatus=pending on SiteCatalog entries (the latter is more extreme). The publishing endpoint OpenMarket/Xcelerate/PrologActions/Publish/<your_delivery_type>/RemoteCall needs to remain accessible from the staging system to delivery, but can be blocked to the outside world.

Like 4.3 you should create rules in your webserver to prevent people from executing commands using CatalogManager, ftcmd etc. Using the csstatus=pending is an extreme method of restricting access to remote calls and should only be used if you can not configure your webserver to restrict access.

5 Additional standard tests

This task is to head off any possible problems that could arise from factors within your control.

5.1 Review filesystem configuration

The filesystem configuration needs to be reviewed to ensure that configuration is appropriate to your needs. You should check with your Operating System Administrator(s) to get finer detail configuration information. Some areas of your file system that you should review are;

Unix/Linux
File Handles
Max Inodes
Permissions
Free Space Considerations

Windows
Permissions
Free Space Considerations

5.2 Review Database configuration and response times

Your database configuration should be reviewed on a regular basis as this is a frequently changing component of your system. When going live you should review the database configuration and address some specific areas such as;

Oracle
Tablespace growth options
Max cursors
Recalculate statistics
Analyze all indexes

MS SQL Server

DB2

Sybase

5.3 Review and assess total number of file-descriptors

You should analyze your operating systems needs to determine if you need to change your file descriptors. This is very important because if you file descriptors are set too low then your system may not be able to open new files or sockets to connect with other Content Servers or write information to disk. Consult your Operating System Administrator for more information on the correct setting of your file descriptors.

Solaris
For solaris, to see and set the filedescriptors type;
ulimit –a

Redhat Linux
For redhat linux to see and set the filedescriptors type;
ulimit –a

Windows
There is no easy way to view and set the filedescriptors for windows, you will need to write a small program to retrieve the file descriptors. More information can be found on Microsoft support site.

5.4 Perform sizing on the java heap size

During your load testing and performance testing you should monitor the JVM heap size to increase or decrease it depending on load and garbage collection. Too much garbage collection could indicate that you're running out of heap and that you should increase it. Many JVMs allow you to write out garbage collection logs that you can use to analyze the heap GC activities.

An example of the switches that you need to add for garbage collects is as follows;
-verbose:gc -XX:+PrintGCTimeStamps -XX:+PrintGCDetails

To adjust your max and min heap size an example of the JVM;
-Xms512m -Xmx1024m

To adjust your perm gen size an example of the parameter is;
-XX:MaxPermSize=128m

5.5 Review Page caching strategy

You should check to see which pages that you have cached and which that you do not. Make sure that you cache pages that very rarely change and do not cache pages that frequently change. Cache pages that are requested a lot. For more information on caching strategies please consult your Professional Services Account Executive.

6 Backup Testing

Every system should be tested for backup and restoration.

6.1 Test database backups

Perform a hot and cold backup of your database aswell as a test export of the schema that Content Server is using. Please follow the procedure that you are planning on following for every day backups.

6.2 Test application backups

Perform a backup of Content Server and the relevant files to Content Server so that you're familiar with all the dependants of Content Server. Remember are the following components to backup;
Content Server install directory
Shared File system
Any other directory referenced within Content Server

Also remember that you should perform this backup when Content Server is not running.

6.3 Test application server backups

Perform a backup of your application server and webcomponents so that you're familiar with the process and location of the files.

6.4 Test disaster recovery

Perform a cold backup of all the above pieces and then restore them on a completely separate server so that you're comfortable with disaster recovery of Content Server.

7 Information transfer

Ensuring that all information is communicated with the rest of the team dealing with FatWire is essential to a successfully rollout.

7.1 Provide go-live implementation plan to support

Go-live plan actions and dates, at minimum a week's notice.

7.2 Document all customizations to product behavior, eg UI elements, modified jars

7.3 Document all known outstanding issues and any action plan for their resolution

7.4 Supply names/contact details of persons responsible for managing the go-live event

7.5 Provide all design documentation describing site design, asset models etc

7.6 Provide architectural map and details such as CS versions, database, webservers, load-balancers, etc.

7.7 Provide an export of database, FatWire directory, Shared Filesystem and web application to support

In order to support your golive process with maximum efficiency you should provide a cold backup of your database schema, FatWire directory, Shared filesystem and web application to FatWire Support so that we can replicate your system inhouse to troubleshoot any golive problems.

7.8 Confirm completion (or otherwise) and results of all items above

Confirm completion (or otherwise) and results of all items above.

Appendix A: Checklist

❑Yes ❑No ❑Documentation Attached	1	Ensure that publishing is working correctly
❑Yes ❑No ❑Documentation Attached	1.1	Analyze Firewall
❑Yes ❑No ❑Documentation Attached	1.2	Ensure that all information published is correctly
❑Yes ❑No ❑Documentation Attached	1.3	Review Page Cache
❑Yes ❑No ❑Documentation Attached	1.4	Review batch host configuration
❑Yes ❑No ❑Documentation Attached	2	Load Testing
❑Yes ❑No ❑Documentation Attached	2.1	Load Test Delivery
❑Yes ❑No ❑Documentation Attached	2.2	Verify cluster configuration
❑Yes ❑No ❑Documentation Attached	2.3	Simulate Editorial Load
❑Yes ❑No ❑Documentation Attached	3	Performance Testing
❑Yes ❑No ❑Documentation Attached	3.1	Performance Test Publishing
❑Yes ❑No ❑Documentation Attached	3.2	Performance Test Content Server
❑Yes ❑No ❑Documentation Attached	3.3	Performance Test additional components eg file system, engage
❑Yes ❑No ❑Documentation Attached	4	Perform a security audit of all external-facing environments
❑Yes ❑No ❑Documentation Attached	4.1	Review Passwords
❑Yes ❑No ❑Documentation Attached	4.2	Harden servlets
❑Yes ❑No ❑Documentation Attached	4.3	Harden non-delivery pages
❑Yes ❑No ❑Documentation Attached	4.4	Protect against remote calls
❑Yes ❑No ❑Documentation Attached	5	Additional standard tests
❑Yes ❑No ❑Documentation Attached	5.1	Review filesystem configuration
❑Yes ❑No ❑Documentation Attached	5.2	Review Database configuration and response times
❑Yes ❑No ❑Documentation Attached	5.3	Review and assess total number of file-descriptors
❑Yes ❑No ❑Documentation Attached	5.4	Perform sizing on the java heap size
❑Yes ❑No ❑Documentation Attached	5.5	Review Page caching strategy
❑Yes ❑No ❑Documentation Attached	6	Backup Testing
❑Yes ❑No ❑Documentation Attached	6.1	Test database backups
❑Yes ❑No ❑Documentation Attached	6.2	Test application backups
❑Yes ❑No ❑Documentation Attached	6.3	Test application server backups
❑Yes ❑No ❑Documentation Attached	6.4	Test disaster recovery
❑Yes ❑No ❑Documentation Attached	7	Information transfer
❑Yes ❑No ❑Documentation Attached	7.1	Provide go-live implementation plan to support
❑Yes ❑No ❑Documentation Attached	7.2	Document all customizations to product behavior, eg UI elements, modified jars
❑Yes ❑No ❑Documentation Attached	7.3	Document all known outstanding issues and any action plan for their resolution
❑Yes ❑No ❑Documentation Attached	7.4	Supply names/contact details of persons responsible for managing the customer go-live
❑Yes ❑No ❑Documentation Attached	7.5	Provide all design documentation describing site design, asset models etc
❑Yes ❑No ❑Documentation Attached	7.6	Provide architectural map and details such as CS versions, database, webservers, load-balancers, etc.
❑Yes ❑No ❑Documentation Attached	7.7	Provide an export of database, FatWire directory, Shared Filesystem and web application to support
❑Yes ❑No ❑Documentation Attached	7.8	Confirm completion (or otherwise) and results of all items above